Contents lists available at Science-Gate

International Journal of Advanced and Applied Sciences

Journal homepage: http://www.science-gate.com/IJAAS.html

Evolution of existing software to mobile computing platforms: Framework support and case study

Adel Alkhalil *

College of Computer Science and Engineering, University of Ha'il, Ha'il, Saudi Arabia

ARTICLE INFO

Article history:
Received 8 September 2020
Received in revised form
23 November 2020
Accepted 25 November 2020

Keywords:
Software maintenance and evolution
Mobile computing
Legacy software
Software architecture
Software modernization

ABSTRACT

Mobile computing as ubiquitous and pervasive technology supports portable and context-aware computation. To date, there exist a significant number of traditional computing systems–running on the web and/or workstation-based platforms–that lack features of mobile computing, including but not limited to ubiquity, context-sensing, and high interactivity. Software that executes on these traditional computing systems is referred to as legacy software that can be upgraded to exploit the features of mobile technologies. However, legacy software may contain critical data, logic, and processes that cannot be easily replaced. One of the solutions is to evolve legacy software systems by (a) upgrading their functionality while (b) preserving their data and logic. Recently research and development efforts are focused on modernizing the legacy systems as per the needs of service and cloud-based platforms. However, there does not exist any research that supports a systematic modernization of legacy software as per the requirements of the mobile platforms. We propose a framework named Legacy-to-Mobile as a solution that supports an incremental and process-driven evolution of the legacy software to mobile computing software. The proposed Legacy-to-Mobile framework unifies the concepts of software reverse engineering (recovering software artifacts) and software change (upgrading software artifacts) to support the legacy evolution. The framework follows an incremental approach with four processes that include (i) *evolution planning*, (ii) *architecture modeling*, (iii) *architecture change,* and (iv) *software validation* of mobile computing software. The framework provides the foundation (as part of futuristic research) to develop a tool prototype that supports automation and user decision support for incremental and process-driven evolution of legacy software to mobile computing platforms.

1. Introduction

Mobile computing empowers its users to exploit portability, context-sensitivity, and enhanced interactivity that is lacking in traditional computing systems (Pejovic and Musolesi, 2015). The increasing growth of mobile computing and rapid adoption of mobile technologies is due to the availability of embedded sensors (device's hardware) that exploit freely available apps (device's software) to communicate with remote servers (device's connectivity) to perform a multitude of tasks on the go (Lane et al., 2010; Campbell and Choudhury, 2012). For example, a mobile device's software (i.e., location tracking service) can utilize the device's hardware (i.e., GPS) to enable portable and context-aware computing to support activities that range from location-driven ride-hailing to digital matchmaking and traffic route planning (Campbell and Choudhury, 2012). As per GSMA's real-time tracker, currently, there are more than 8.9 billion mobile device connections that represent approximately five times faster growth of mobile connections than the human population (Intelligence, 2016). However, to support the sustainable growth of mobile computing technologies, the traditional computing paradigms–software systems that run on workstation or web platforms–must be modernized so they can execute on mobile platforms and benefit from mobile computing technologies (Foss and Wong, 2004; Sørensen et al., 2003). The software that runs on traditional computing systems is referred to as legacy software that may be developed/based on

* Corresponding Author.
Email Address: a.alkalel@uoh.edu.sa
https://doi.org/10.21833/ijaas.2021.03.013
Corresponding author's ORCID profile:
https://orcid.org/0000-0003-3135-9174

conventional or outdated technologies that need an upgrade (in the context of mobile computing), however; legacy software may contain critical data, logic, and processes that are fundamental to the operations of a particular domain (Khadka et al., 2013; Jamshidi et al., 2013; Bennett, 1995). For example, an online product selling store contains all the product-specific data and logic to provide a web-based interface to view, select and order the products. However, such a system lacks context-sensitivity (e.g., dynamic location information) and portability (i.e., computation on the go) to offer customized offers and recommendations based on contextualized information supported by mobile technologies. With the ever-increasing adoption of mobile technologies and the existence of legacy systems, there is a need to develop solutions that enable or enhance the modernization of legacy software so it can benefit from the features of mobile computing (Sørensen et al., 2003).

Software evolution refers to a systematic change implementation in the structure and behavior of the existing software as per the changes in the system's requirements and operational environments (Mens, 2008). The terms evolution, migration, and modernization are virtually synonymous, and all refer to adaptive change (Williams and Carver, 2010). We explicitly use the term evolution that refers to evolving an existing software as per the new requirements or the needs for upgraded functionality or quality of the software. For example, Jamshidi et al. (2013) presented a survey that highlights solutions that support structural and behavioral evolution of legacy or existing software systems to cloud-based services that can be managed via cloud computing technologies and infrastructure. The evolution of software-intensive systems can be categorized into different types, namely perfective, adaptive, corrective, and preventive evolution, with further details of each type presented in Williams and Carver (2010). In this research, we focus on supporting adaptive evolution that refers to upgrading or adapting a software's structure and behavior as per new requirements or needs of the new platform. Specifically, we aim to propose and develop a process-based and increment-driven evolution of the legacy software systems to portable, context-aware, and interactive mobile computing platforms. However, to support such evolution, the primary challenge lies with exploiting a systematic reengineering approach (i) that upgrades software functionality but (ii) preserves the critical data, logic, and processes that cannot be altered to ensure correct functionality and quality of the software (Ahmad and Babar, 2014). For example, when a traditional (workstation-based) geographical information system is migrated to mobile platform(s), it contains the same geographical data and relies on the same algorithms along with additional features of portability, interactivity, context-sensing, and location-awareness to capture precise geographical information on the go (Foss and Wong, 2004). However, the evolution of such legacy software to mobile platforms can be a daunting task due to many issues that include but are not limited to platform compatibility, evolved requirements, and restrictions due to resource-poverty (available power, storage, processor) of mobile devices (Sørensen et al., 2003). In order to overcome the technical challenges detailed above, as well as to avoid managerial issues involved in the migration process, solutions are needed that support a systematic and process-driven (automation of) legacy software migration to mobile computing platforms (Ahmad and Babar, 2014).

In recent years, the research and development on software evolution have primarily focused on the migration of existing or legacy systems to service-oriented-and cloud-based systems (Khadka et al., 2013; Jamshidi et al., 2013). There is a lack of efforts that utilize the theory and practices of software evolution to enable the modernization of legacy software as per the needs of the mobile computing platform. The existing research on software evolution to mobile computing focuses on upgrading the legacy software for mobile devices (Foss and Wong, 2004; Sørensen et al., 2003). However, such evolution efforts are complex, time-consuming, and manual that hinder the process for evolution. The state-of-the-art research suggests the need for a systematic approach that relies on reusable knowledge and practices, prototypes, and tool support along with human intervention and decision support to guide and execute legacy software evolution to mobile computing (Matthiesen and Bjørn, 2015).

Overview and Implications of Solution: We propose to integrate software reverse engineering and software change to develop a framework that supports a systematic evolution of legacy to mobile and provides a foundation for automation and tool support. Specifically, we propose a framework named Legacy-to-Mobile that relies on (a) software reverse engineering to recover the artifacts (e.g., code or architecture) of legacy software and (b) software change management to upgrade the artifacts of legacy software as per requirements of the mobile computing platform. A high-level view of the proposed solution is provided in Fig. 1. Fig. 1 highlights that the proposed Legacy-to-Mobile solution framework comprises of four processes, namely (i) *Planning the evolution*, (ii) *Modeling the artifacts of legacy software*, (iii) *Transformation of the legacy artifacts to mobile computing artifacts*, and (iv) *Refinement of the evolved software* based on validations to ensure the desired quality and functionality of the mobile software. The role of user decision (human supervision) and tool support (semi-automation) for the evolution process is central in the proposed framework. The solution must consider and support the structural and behavioral changes in the legacy software during evolution to the mobile (Mens, 2008; Williams and Carver, 2010). Moreover, the solution must support the performance of the evolved software in the context of resource-limited mobile computing

devices (Pejovic and Musolesi, 2015; Lane et al., 2010). Based on the solution proposal, we lost the core contributions as:

- Exploiting the theory and practices of software evolution to enable the evolution of legacy software towards portable and context-sensitive mobile computing applications.
- Utilizing architectural abstraction to abstract the complexities to support the evolution of legacy software to drive the evolutionary processes that are lacking in the existing solutions.
- Developing an evolution process and framework that provides a frame of reference to develop an appropriate tool and human decision support for process automation and customization.

The proposed research is based on two different phases. The first phase involves the development of a framework (scope of this paper) that serves as a foundation for the second phase to enable automation and customization of the process (needs for future research).

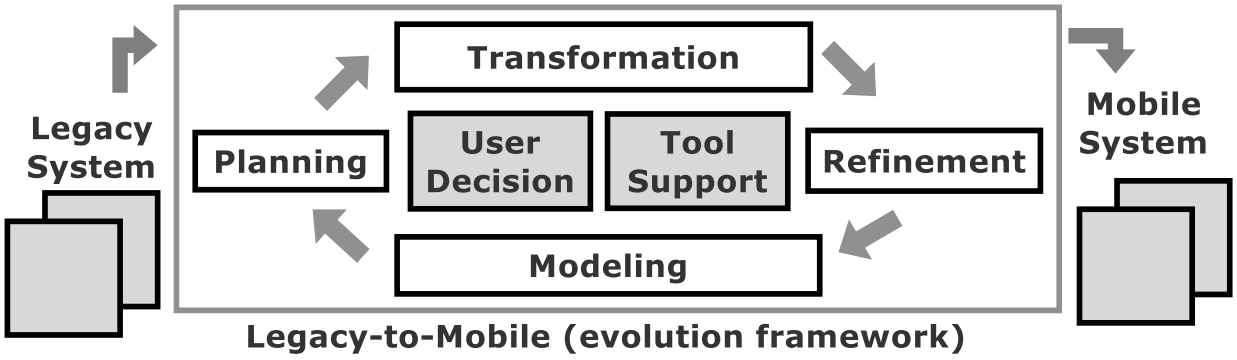

Fig. 1: Overview of the proposed solution

Paper Organisation: Related research is presented in Section 2. A research method is detailed in Section 3. Solution overview as a proposed framework is detailed in Section 4. Case study and solution validation are in Section 5. Section 6 provides conclusions with limitations and dimensions of future research.

2. Related research

Legacy software automates and performs critical operations for an organization/enterprise, and such software has long life-spans, it does not emerge from scratch, it can be expensive or critical to replace and requires frequent maintenance and evolution to continue software operations (Bennett, 1995). The research and development on the evolution or modernization of legacy systems date back to 1970s when the Lehman's laws of software evolution urged the need to frequently evolve existing or legacy software to support its operations (Ahmad and Babar, 2014). We present existing work that evolves legacy software systems to modern computing platforms. Specifically, we discuss the existing research and development on legacy migration to software services and cloud (in Section 2.1) that help us to position the needs for legacy migration to mobile computing (in Section 2.2) in the context of exiting research.

2.1. Evolution of legacy systems to modern computing platforms systems

Recently a lot of research and development is done on the evolution of legacy software systems to modern computing platforms that mainly include service-oriented systems, cloud-based platforms, or software product lines (Khadka et al., 2013; Jamshidi et al., 2013; Assunção et al., 2017). Such an evolution can support the efficiency and economy of software operations. For example, most recently, the US department of defense (US DoD) has been able to achieve IT-specific process efficiency, including enhanced software security by modernizing their legacy enterprise mail system as a cloud-based email system. Moreover, by means of software modernization, US DoD has achieved an operational economy (i.e., saving a cost of $100 m approx. per annum) for running their IT systems on cloud servers. In this context, a relevant example is the evolution of Oracle client-server architecture to a cloud computing platform (Laszewski and Nauduri, 2011).

Evolving Legacy Software to Service-driven Architectures: In the context of legacy software evolution, Khadka et al. (2013) presented a survey of existing research focusing on the solutions of legacy to Service Oriented Architecture (SOA) evolution. The results of the SLR provide a repository of research solutions to support evolution. In Winter and Ziemann (2007), the horseshoe model for incremental evolution of legacy software to SOA is presented. The horseshoe model provides a generic framework that requires processes, patterns, and tool support for legacy evolution to SOAs.

Modernization of Legacy Software for Cloud-based Platforms: In Jamshidi et al. (2013), the authors present a Cloud Reference Migration Model (Cloud-RMM) that supports a process for migration of legacy software to cloud computing systems. Cloud-RMM is developed based on analyzing the existing solutions and consolidating them in a process for migration to the cloud. Some of the

recent projects (Mohagheghi and Sæther, 2011; Alonso et al., 2013; Frey and Hasselbring, 2011) support migration of legacy systems to cloud-based environments (IaaS) and (PaaS). In recent years, some internationally collaborated on research and development projects such as *REMICS* (Mohagheghi and Sæther, 2011), *CloudMig* (Alonso et al., 2013), and *ARTIST* (Frey and Hasselbring, 2011) to support a systematic approach for codifying the legacy systems.

Legacy Migration to Software Product Lines (SPLs): In Assunção et al. (2017), the authors performed a mapping study to investigate the existing research on the re-engineering of existing systems to SPLs. The mapping study has identified open challenges and areas for future research that highlight the need for tool prototypes that automate the migration.

2.2. Legacy software to operate on mobile computing platforms

In contrast to the research and development on legacy software evolution to service-based (Jamshidi et al., 2013) and cloud-driven platforms (Khadka et al., 2013), there is much less research on legacy evolution to mobile. One of the pioneering works on legacy software evolution towards mobile computing platforms is presented in Pope (1996) that enabled the migration of software applications from a resource-rich work-station based computing environment to more self-reliant mobile computers. In the last decade and specifically beyond the year 2010, the rapid emergence and adoption of mobile devices, affordable networking, and free software apps have made mobile computing an ideal platform adopted by legacy software as part of software modernizations. The primary motivations for legacy modernization as per mobile platforms are to achieve context-sensitivity and enhanced usability of the existing systems. For example, the research in Foss and Wong (2004) supported the migration (by re-engineering the source code) of a Windows-based software running on a desktop machine towards a Palm OS-based app running on Personal Digital Assistant (PDA). In a similar study (Fan and Wong, 2016), the authors present a solution that enables the modernization of the user interfaces (UIs) of legacy software by evolving them to interactive and touch-based interfaces to facilitate the ease of use for legacy software. In terms of the partial tool support for legacy evolution to mobile, the only work is Williams and Carver (2010) and Canfora et al. (2004), which enabled the migration of desktop-based applications to hand-held device platforms. The existing research lacks a systematic approach and solution(s) to enable legacy software evolution to mobile computing platforms. In order to support the current and future research, the role of patterns, processes (as reusable knowledge and practices), along with frameworks and tool support, is vital to support the evolution.

Conclusive Summary: After presenting an overview of the research state-of-the-art on legacy software evolution, we now present a summary of the comparison between existing research and proposed solution as in Table 1. We have used four-element criteria with source, and target represents the existing (evolution from) and modernized (evolution to) platform. Moreover, tool support and the human decision represents support for automation and user intervention in the evolution process. Based on the data in Table 1, we can conclude that our proposed solution is conceptually similar to Winter and Ziemann (2007) that followed an incremental and process-driven approach for the evolution of legacy software to the mobile computing platform. Based on the approach in Alonso et al. (2013) and Frey and Hasselbring (2011), we aim to provide tool-supported automation and human decision-driven supervision to execute and guide the software evolution. Moreover, in contrast to the existing research of legacy modernization, we focus on the evolution of legacy software to mobile computing platforms that are currently lacking in the existing research.

3. Research methodology

We now present the research methodology we adopted to conduct this research. The methodology is detailed based on illustrations in Fig. 2. The section also introduces concepts that will be used throughout the paper.

3.1. Outlining the research hypothesis

As in Fig. 2, the first step of the methodology is to outline the research hypothesis. Research hypothesis provides us the foundation to develop the process and validate the hypothesis based on a case study based approach. We outline the research hypothesis as:

A systematic identification of the literature helps to empirically design the process that supports an incremental evolution of existing software systems to mobile computing platforms.

After outlining the hypothesis, we now conduct this research into two distinct phases, as per Fig. 2, with each phase detailed below.

3.2. Phase I: Conducting the systematic mapping study

The first phase involves a systematic literature review that is conducted by following the guidelines and recommendations for undertaking the systematic mapping studies (Petersen et al., 2008). Systematic mapping study allows a systematic identification, analysis, and review of the existing research that provides the strengths, limitations, and research gaps in the topic under investigation.

Table 1: Comparison between existing solutions and proposed research

Solution Reference	Source	Target	Tool Support	Human Decision
Evolution of Legacy Software to Service-Oriented Architectures				
(Khadka et al., 2013; Winter and Ziemann 2007)	Legacy Enterprise Software	SOA	No	No
Modernization of Legacy Software to Cloud-based Platforms				
(Jamshidi et al., 2013; Mohagheghi and Sæther, 2011; Alonso et al., 2013; Frey and Hasselbring, 2011)	Legacy Software	Cloud Computing (SaaS, PaaS, IaaS)	Yes (Alonso et al., 2013; Frey and Hasselbring, 2011)	Yes (Alonso et al., 2013)
Migration of Legacy Software to Software Product Lines				
(Assunção et al., 2017)	Legacy Software	Software Product Line	No	No
Migration of Legacy Software to Mobile Computing Platforms				
Proposed Solution	Legacy Software	Mobile Platform	Yes (Future Work)	Yes (Future Work)

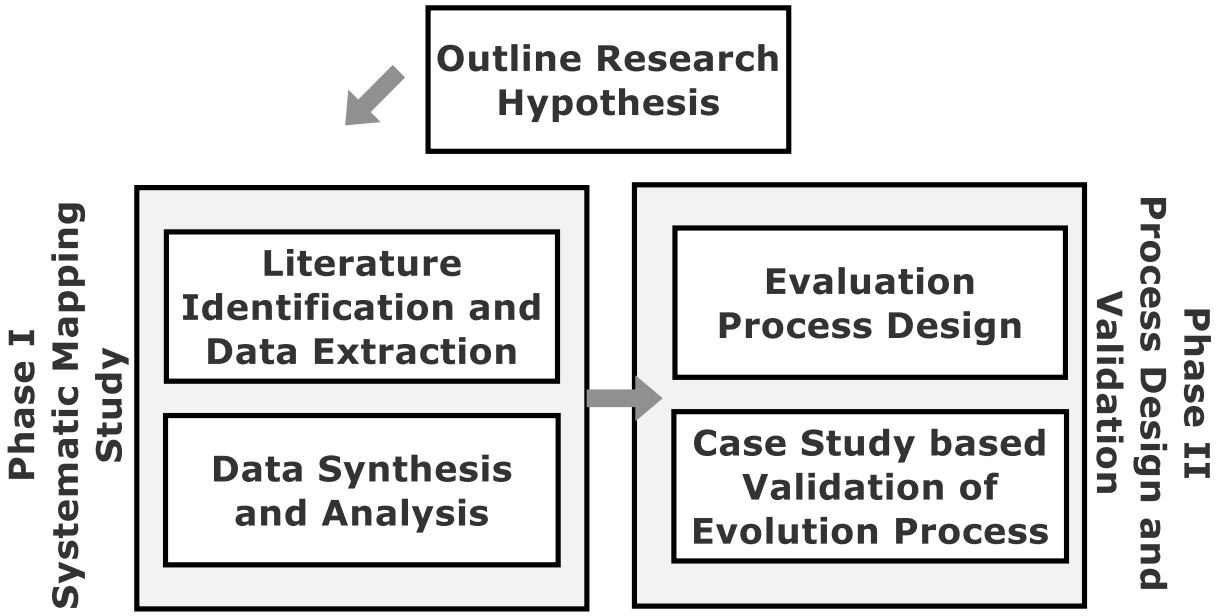

Fig. 2: Overview of the research methodology

Literature Identification and Data Extraction: As the first step, we identified the literature in terms of published research on software evolution for mobile computing. Literature identification is followed by a qualitative selection of quality research to exclude any research studies that do not meet a pre-defined quality criterion. Once the research studies were selected, the next step involved data extraction from the identified studies. Data extraction is done by locating the relevant information in a given research study. For example, type of evolution as design-time evolution or runtime evolution, type of software that needs to be evolved, i.e., workstation-based system or web portal. The extracted information is maintained for data synthesis in the next step.

Data Synthesis and Analysis; Data synthesis step involves a systematic mapping of the extracted data. The main purpose of data synthesis is to find recurring themes of research in the existing literature. The recurring themes help us to classify the research based on the focus and contributions of the research in terms of overlapping or distinct research contributions. After data is synthesized, in this phase, the last step is data analysis. The data analysis step primarily focuses on mapping the existing research contributions, their strengths, and limitations, along with the research gap that needs a solution to address them. Extended details of the results of the mapping study are published in Sultan (2019).

3.3. Phase II: Designing the validating the evolution process

After conducting the mapping study, Phase II involves designing and validating the evolution process. Both of these steps are detailed below:

- Design the Evolution Process: The evolution process aims to address the shortcoming of the existing research (Sultan, 2019) and outlines a process-centric and incremental evolution of the existing software to mobile computing platforms. The process is divided into different activities that are illustrated in Fig. 3. Fig. 3 demonstrates four distinct activities that constituent the evolution process. These activities are referred to as Planning, Modeling, Transformation, and Evaluation. The evolution process and all its underlying activities are detailed in Section 4.

- Case Study-based Validation of the Evolution Process: The last step in the research methodology, as per Fig. 2, is case study based validation of the evolution process and outlined hypothesis (Section

3.1). Case study-based validation allows us to develop some use cases for the evolution. Use-cases provide scenario-based practical demonstration and evaluation of the process. Case study based validation of the process and hypothesis are detailed in Section 5.

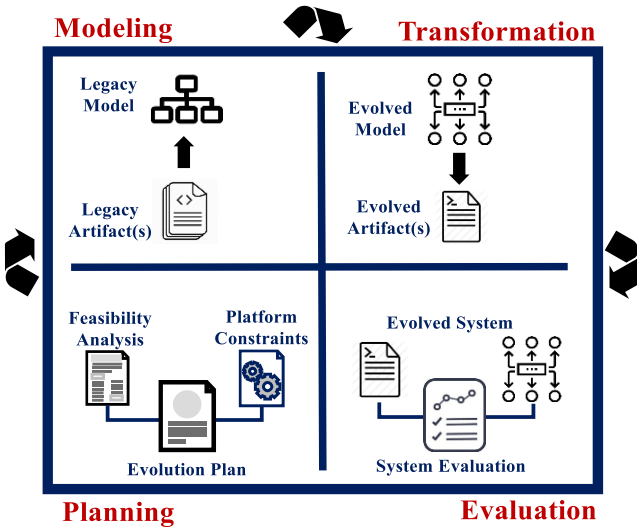

Fig. 3: Process view for software evolution to mobile computing

4. Process-centric solution for legacy-to-mobile evolution

We now present the framework that is named as Legacy-to-Mobile that enables an incremental evolution (i.e., step-by-step change implementation) of legacy software to the mobile computing platform. A high-level view of the proposed framework is presented in Fig. 2 that highlights processes and activities that enable software evolution. Incremental evolution refers to step-wise change management in existing software to support its incremental evolution. We now present the framework in terms of the framework processes and activities along with the income and outcome of each process that drives legacy evolution.

The processes refer to *what needs to be done*? While the activities highlight *how it is to be done?* As illustrated in Fig. 2. We have provided a summary of all the processes and their underlying activities in Table 2. Table 2 also represents a structured catalog to capture and presents the process activities, incomes, and outcomes of each process, along with process automation and supervision. We now discuss each individual process of the framework based on the illustrations from Fig. 2 and a summary of processes in Table 2.

4.1. Process I: Planning for legacy evolution

The first process in legacy to mobile evolution, as presented in Fig. 4, relates to the planning for systematic management and execution of changes in the legacy software. The evolution plan guides the further processes. This process aims to investigate: How to develop an evolution plan that accommodates the needs of legacy evolution and feasibility for executing such evolution to support an incremental change of legacy software as per the needs for mobile computing platforms.

Process Income and Outcome: As the initial process of the framework, the planning process does not require any input to the process. Instead, the needs and motivations for legacy evolution can be considered as the pre-requisites for evolution planning. The outcome of the process is an evolution plan that acts as a blueprint and a documented reference for the stakeholder before proceeding further with the evolution.

Activity A-Feasibility Study: The first activity during evolution planning is to conduct a feasibility study that aims to assess the costs and benefits of the desired evolution. Moreover, the feasibility study investigates if it would be practically feasible to evolve the legacy system. The feasibility study must be conducted by the stakeholders (e.g., users, owners, engineers) of the software system that is the candidate for the migration.

Activity B-Platform Analysis: After the feasibility study, the next activity relates to analyzing the source (legacy) and the target (mobile) platform. The platform analysis is critical to understand if the legacy platform has some constraints that can hinder the evolution. Moreover, the analysis for the target platform highlights that the migrated software must be efficient in terms of exploiting the resource of resource-constrained mobile devices.

Process Automation and Supervision: Evolution planning is a manual process that requires active human intervention (stakeholders' needs and requirements) to guide the legacy evolution. Active human supervision and intervention are required in order to support the process execution.

4.2. Process II: Modeling the legacy software system

After the planning, the next process is the modeling (i.e., structural representation) of the legacy system that needs to be transformed. Modeling provides an overall structural view of the legacy software as a blueprint of the system. The model can abstract the implementation-specific complexities of the software with a design or architectural view that presents a graphical (high-level) view of the software. Therefore, the model as a high-level graphical representation of the legacy software allows us to analyze and transform the system at a higher level of abstractions (Ahmad et al., 2019; 2014).

Process Income and Outcome: The income to this process is the specification (source code or design) of the legacy software system that is the candidate for evolution. The outcome is the model (high-level view of the system) in terms of the design or architecture of the legacy system that needs to be evolved.

Activity A–Specification Analysis: The first activity of this process is to analyze the specifications of the

legacy software. The analysis allows us to view and analyze the system representation that can be represented as a model for its evolution.

Activity B–Structural Constraints: In the represented model, we need to identify the constraints that may be enforced on the structure, and that needs to be preserved. Structural preservation is also vital to ensure the structural integrity of the source model.

Process Automation and Supervision: Legacy system modeling is a manual process that means the model can be generated from the source code specifications in an automated way. However, user intervention and supervision are required to ensure that the represented model represents correct specifications.

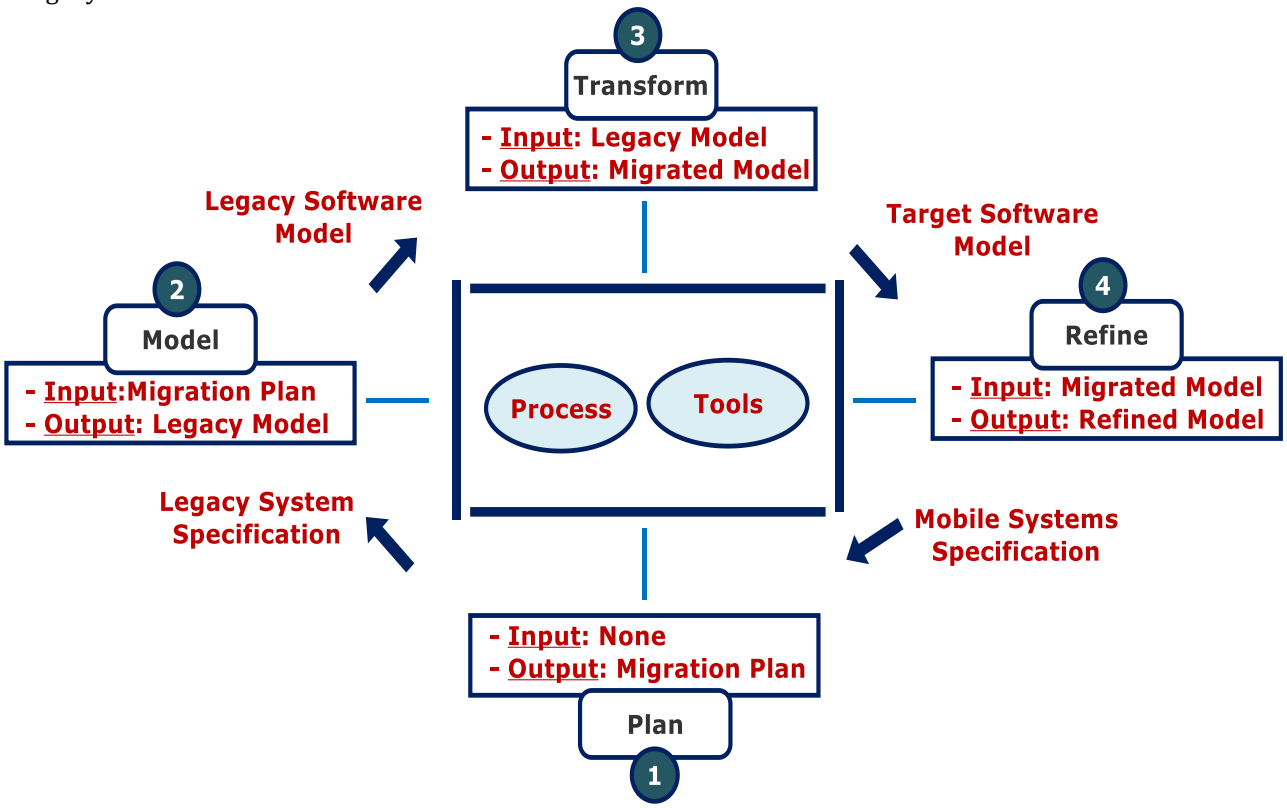

Fig. 4: Overview of the legacy-to-mobile framework for incremental evolution of legacy software to the mobile computing platform

4.3. Process III: Transformation of legacy software system

After modeling the legacy software, the third phase of the process corresponds to the transformation of the legacy software as a mobile-enabled application. Software transformation refers to the addition, removal, or modification of software components and modules to enable software evolution. Transformation-driven evolution of the existing software enables and incremental evolution of the software as per new requirements and business needs.

- **Process Income and Outcome:** The income to this activity is the legacy software model that needs to be evolved. The outcome is the transformed model as a mobile-enabled software application.
- **Activity A–Legacy Modeling:** is achieved through a high-level design or architectural representation of the existing software. High-level modeling complex details with a global view of the system with software represented as architectural components and connectors.
- **Activity B–Legacy Transformation:** Legacy transformation supports change management in the architectural model to support its evolution.
- **Process Automation and Supervision:** Legacy transformation is a semi-automated process where the software model is transformed in an automated way. Human supervision is required to ensure consistency of transformation and ensuring no violation in the transformed structure of the software.

4.4. Process IV: Refinement of the legacy software system

The last process is about the refinement and validation of the transformed software. Validation ensures that the evolved mobile application provides acceptable performance. Specifically, during validation, we need to ensure the accuracy of predictions, computation efficiency of the software when it is being executed on resource-constrained mobile devices with limited computation and processor resources.

- **Process Income and Outcome:** The income to the process is the transformed software. The outcome is the validated software in terms of its accuracy of predictions and computational efficiency of the evolved software.
- **Activity A–Validating the Prediction Accuracy:** The first activity is to measure the accuracy of prediction that is analyzed through precision and recall of the recommendations.
- **Activity B–Ensuring Computation Efficiency:** We measure the CPU usage of the resource-constrained mobile devices when the device is operating and when it is in the idol.
- **Process Automation and Supervision:** Validation is a manual activity where data generated from the evolved software is analyzed against the benchmark to assess the efficiency and accuracy of the system.

5. Case study-based validation: Software evolution for mobile computing

We now present a case study based validation of the evolution process. Case-study based approach is driven by scenarios that help us to demonstrate the evolution process and also perform its validations. Case study and process validation are presented in dedicated subsections below. Specifically, the case study for software evolution is in Section 5.1, whereas validation of the evolved software is presented in Section 5.2.

5.1. Case study for software evolution to mobile computing

The case study for software evolution to mobile computing is being adopted from the smart city market scenario in Khan et al. (2020) and Alreshidi and Ahmad (2019), as illustrated in Fig. 5. In Fig. 5a, we show the existing software that is a web-based system for online product selling and product recommendations. The web-based system has the following shortcomings:

- Lack of Context Sensitivity: The existing system lacks context-sensitivity such as the current location of the user, time of the day, and geo-proximity of the user to a particular marketplace. Due to a lack of context-sensitivity, the system is unable to provide context-aware recommendations to the end-users.
- Lack of Portability: The existing web-based system lacks portability and requires execution on typical workstations to offer proper functionality.

Table 2: Overview of the framework process and activities along with process incomes and outcomes

Framework Processes	Process Activities	Process Income	Process Outcome	Process Automation (Tool Support)	Process Supervision (Human Decision)
Process I Evolution Planning	- Perform Trade-off Analysis - Identify Level of Evolution	None	Evolution Plan - Motivations - Challenges - Level of Evolution	No	Yes
Planning for the evolution of the legacy software by taking into consideration the source and target platforms, benefits, and limitations along with the required efforts for the solution.					
Modeling	- Specification Analysis - Structural Constraints	Legacy Software Specifications	Legacy Software Model	Yes	Yes
Modeling of the legacy software to understand the overall structure of the legacy software and identify areas that require structural changes during evolution.					
Transformation	- Change Implementation - Property Preservation	Legacy Software Model	Target/Evolved Software Model	Yes	Yes
Transformation of the legacy software structure by means of change implementation to evolve it as per the requirements of the new platform/software.					
Refinement	- Consistency Conformance - Structural Optimizations	Target/Evolved Software Model	Refined Software Model	No	Yes
Refinement of the transformed software by means of optimizations to support both the desired functionality and quality of the evolved software.					

Due to the above two limitations, there is a need for the evolution of the existing web-based product recommender systems as context-sensitive mobile recommender systems. The evolved system, as in Fig. 5b, is a mobile-enabled application that supports portability, context-sensitivity, and enhanced interactivity. As illustrated in Fig. 5b, the evolved application takes into account the user's location, preferences, and historical data to provide context-sensitive recommendations to the user. It is vital to mention that the evolution of existing systems (Fig. 5a) to the mobile system (Fig. 5b) is enabled through the evolution process (Section 4, Fig. 4). We now present validation of the evolved software in terms of (i) accuracy of system recommendations of the items of interests and (ii) computation efficiency of the system on resource-constrained mobile devices.

Fig. 5: (a) Web-based Online Portal (before evolution) and (b) Mobile-based system (after evolution)

5.2. Validation of the evolved mobile system

In this section, first, we present system accuracy for recommendations that are measures based on precision and recall of the recommendations. We then present the computation efficiency of the system when it is being executed on resource-constrained mobile devices.

5.2.1. Precision and recall of recommendations

The precision of the recommended measures accuracy of the recommended items, whereas recall of the recommendation estimates the ratio of correct recommendations from total recommendations by the system. Both the precision and recall are illustrated in Fig. 6 and derived based on a minimum of 50 trials from 3 different groups of users. Evaluation results suggest a satisfactory level of precision and recall such that precision values vary from 35 to 46 in 50 trials, whereas the recall value varies between 22 to 42 in different cases. Based on the results of validation, we can conclude that the evolved software has fair accuracy of the recommended items. However, diverse datasets with more rigorous validation are required to further enhance the accuracy of recommendation that is considered as part of future work.

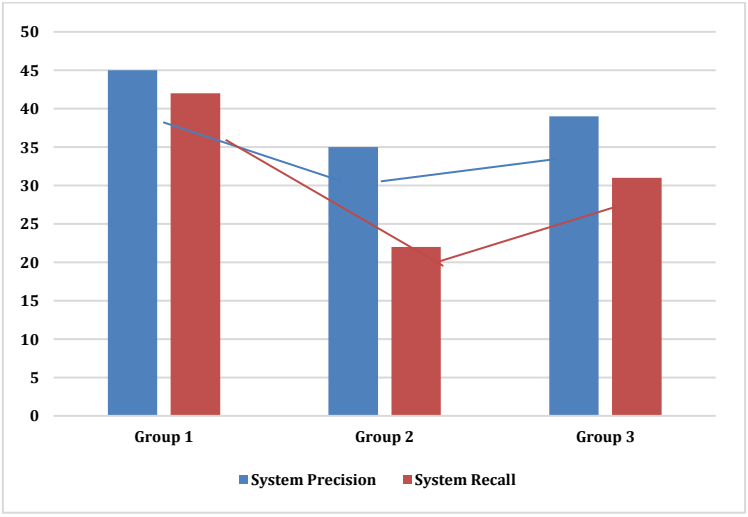

Fig. 6: Overview of the system accuracy for recommendation

5.2.2. Computation efficiency of the evolved software

It is vital to measure the computation efficiency of the evolved software. Measuring computation efficiency is vital to ensure that the evolved software functions properly when migrated from resource-rich workstations to resource-constrained mobile devices. We highlight the results of the device's CPU utilization in Fig. 7. Fig. 7 is based on 100 trials to measure how the execution of the evolve software impacts the CPU. The data in Fig. 7 shows satisfactory performance with the device using from 2 to 5 percentage of the overall CPU. This means that when the mobile application is running and active, it can take up to 5% of the CPU. Alternatively, when the application is inactive (running in the background), it takes only 2% of the CPU. Further trials are needed to validate the CPU utilization on different devices and different mobile platforms.

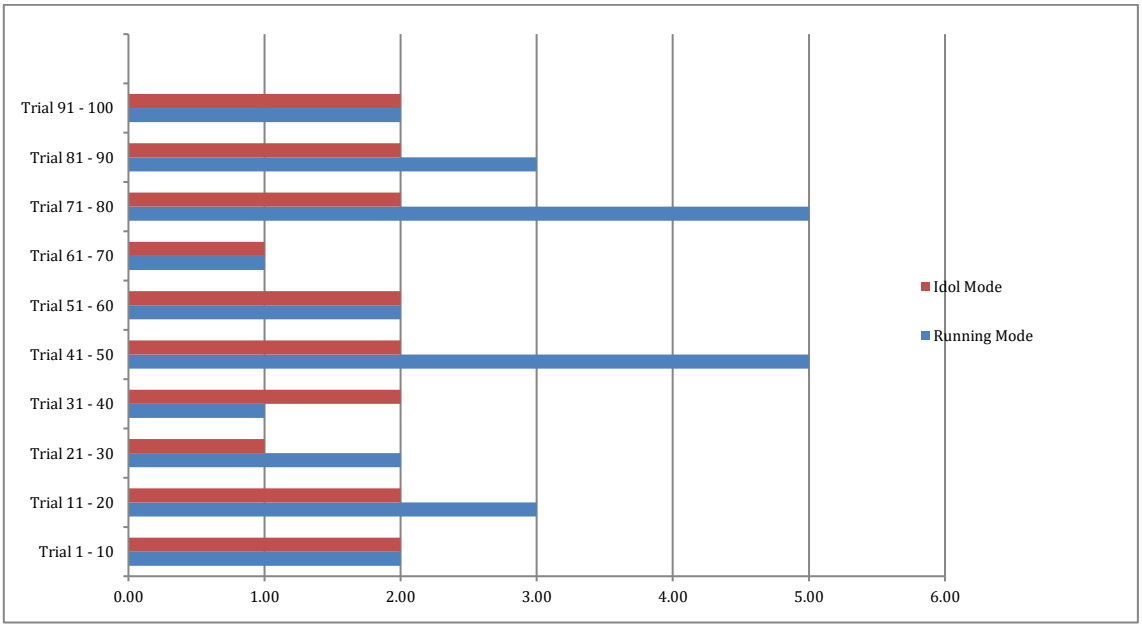

Fig. 7: Overview of the system computation efficiency

6. Conclusions, limitations, and future research

The proposed solution aims to support the research and solutions based on software evolution. More specifically, the solution exploits the 'law of continuing change' to support the evolution-driven modernization of the legacy software systems so they can be deployed and executed on mobile computing platforms. Since the last decade, the research on software evolution has started to get significant attention, also highlighting some of the critical challenges that relate to the planning, execution, and automation of software evolution. Automation of software evolution is a significant challenge as it requires tool support along with appropriate user supervision and decision support to execute a significant number of changes in an efficient way. To date, there is no research and development on tool support for the modernization of the legacy systems to other (modernized) computing platforms.

Potential Limitations of the Research: We also discuss two potential limitations of the proposed solution. Addressing these limitations also reflect possible dimensions of the future work.

Limited Validation with Single Case Study: Currently, the proposed evolution process is validated based on a single case study as in Fig. 5. The case study-based approach provides scenarios for demonstrative validation of the core concepts of solution. However, only a single case study could limit the generalization of the results and applicability of the solution. Therefore, there is a need for further case studies and evolution scenarios to validate the solution in a different context.

Diversity of Scenarios for Validating Accuracy and Computation Efficiency: The accuracy or recommendations and computation efficiency of the mobile system are evaluated based on a total of 50 trials organized into 03 distinct groups as in Fig. 6 and Fig. 7. The results help us to validate the performance and functionality of the evolved software. However, there is a need for more trials and more groups with different data sets to avoid any bias and limitations of validations. In the future, we plan to extend the validation based on new data and case studies for evolution.

Dimensions of Future Research: The scope of the existing solution is an incremental evolution of existing software towards a portable mobile-enabled system. In the future, we aim to extend the scope of existing work to emerging domains such as big data and block-chain systems. This means that instead of throwing away existing software, we can incrementally evolve them as big data analytics and block-chain based systems. Such a solution would

need case studies from legacy systems and customization of the evolution process (Fig. 4) that can address the emerging challenges of evolving legacy software towards modernized computing platforms such as big data and block-chain systems. Moreover, mobile security is another concern that needs to be addressed as part of future work (Sajjad et al., 2018; Ahmad et al., 2019).

Compliance with ethical standards

Conflict of interest

The author(s) declared no potential conflicts of interest with respect to the research, authorship, and/or publication of this article.

References

Ahmad A and Babar MA (2014). A framework for architecture-driven migration of legacy systems to cloud-enabled software. In the Proceedings of the WICSA 2014 Companion Volume, Association for Computing Machinery, Sydney, Australia: 1-8. https://doi.org/10.1145/2578128.2578232

Ahmad A, Alkhalil A, Altamimi AB, Sultan K, and Khan W (2019). Modernising legacy software as context-sensitive and portable mobile-enabled application. IEEE Computer Society, Washington, USA.

Ahmad A, Jamshidi P, and Pahl C (2014). Classification and comparison of architecture evolution reuse knowledge-A systematic review. Journal of Software: Evolution and Process, 26(7): 654-691. https://doi.org/10.1002/smr.1643

Ahmad A, Malik AW, Alreshidi A, Khan W, and Sajjad M (2019). Adaptive security for self-protection of mobile computing devices. Mobile Networks and Applications, 1-20. https://doi.org/10.1007/s11036-019-01355-y

Alonso J, Orue-Echevarria L, Escalante M, Gorroñogoitia J, and Presenza D (2013). Cloud modernization assessment framework: Analyzing the impact of a potential migration to Cloud. In the IEEE 7th International Symposium on the Maintenance and Evolution of Service-Oriented and Cloud-Based Systems, IEEE, Eindhoven, Netherlands: 64-73. https://doi.org/10.1109/MESOCA.2013.6632736

Alreshidi A and Ahmad A (2019). Architecting software for the internet of thing based systems. Future Internet, 11(7): 153. https://doi.org/10.3390/fi11070153

Assunção WK, Lopez-Herrejon RE, Linsbauer L, Vergilio SR, and Egyed A (2017). Reengineering legacy applications into software product lines: A systematic mapping. Empirical Software Engineering, 22(6): 2972-3016. https://doi.org/10.1007/s10664-017-9499-z

Bennett K (1995). Legacy systems: Coping with success. IEEE Software, 12(1): 19-23. https://doi.org/10.1109/52.363157

Campbell A and Choudhury T (2012). From smart to cognitive phones. IEEE Pervasive Computing, 11(3): 7-11. https://doi.org/10.1109/MPRV.2012.41

Canfora G, Di Santo G, and Zimeo E (2004). Toward seamless migration of Java AWT-based applications to personal wireless devices. In the 11th Working Conference on Reverse Engineering, IEEE, Delft, Netherlands: 38-47. https://doi.org/10.1109/WCRE.2004.38

Fan X and Wong K (2016). Migrating user interfaces in native mobile applications: Android to iOS. In the IEEE/ACM International Conference on Mobile Software Engineering and Systems, IEEE, Austin, USA: 210-213. https://doi.org/10.1145/2897073.2897101
PMCid:PMC4778641

Foss A and Wong K (2004). On migrating a legacy application to the palm platform. In the 12th IEEE International Workshop on Program Comprehension, IEEE, Bari, Italy: 231-235. https://doi.org/10.1109/WPC.2004.1311065

Frey S and Hasselbring W (2011). The clouding approach: Model-based migration of software systems to cloud-optimized applications. International Journal on Advances in Software, 4(3 and 4): 342-353.

Jamshidi P, Ahmad A, and Pahl C (2013). Cloud migration research: A systematic review. IEEE Transactions on Cloud Computing, 1(2): 142-157. https://doi.org/10.1109/TCC.2013.10

Khadka R, Saeidi A, Idu A, Hage J, and Jansen S (2013). Legacy to SOA evolution: A systematic literature review. In: Ionita AD, Litoiu M, and Lewis G (Eds.), Migrating legacy applications: challenges in service oriented architecture and cloud computing environments: 40-70. IGI Global, Pennsylvania, USA. https://doi.org/10.4018/978-1-4666-2488-7.ch003

Khan A, Ahmad A, Rahman AU, and Alkhalil A (2020). A mobile cloud framework for context-aware and portable recommender system for smart markets. In: Mehmood R, See S, Katib I, and Chlamtac I (Eds.), Smart infrastructure and applications: 283-309. Springer, Cham, Switzerland. https://doi.org/10.1007/978-3-030-13705-2_12

Lane ND, Miluzzo E, Lu H, Peebles D, Choudhury T, and Campbell AT (2010). A survey of mobile phone sensing. IEEE Communications Magazine, 48(9): 140-150. https://doi.org/10.1109/MCOM.2010.5560598

Laszewski T and Nauduri P (2011). Migrating to the cloud: Oracle client/server modernization. Elsevier, Amsterdam, Netherlands. https://doi.org/10.1016/B978-1-59749-647-6.00001-6

Matthiesen S and Bjørn P (2015). Why replacing legacy systems is so hard in global software development: An information infrastructure perspective. In the 18th ACM Conference on Computer Supported Cooperative Work and Social Computing, Association for Computing Machinery, Vancouver, Canada: 876-890. https://doi.org/10.1145/2675133.2675232

Mens T (2008). Introduction and roadmap: History and challenges of software evolution. In: Mens T, and Demeyer S (Eds.), Software evolution: 1-11. Springer, Berlin, Germany. https://doi.org/10.1007/978-3-540-76440-3_1

Mohagheghi P and Sæther T (2011). Software engineering challenges for migration to the service cloud paradigm: Ongoing work in the REMICS project. In the IEEE World Congress on Services, IEEE, Washington, USA: 507-514. https://doi.org/10.1109/SERVICES.2011.26

Pejovic V and Musolesi M (2015). Anticipatory mobile computing: A survey of the state of the art and research challenges. ACM Computing Surveys (CSUR), 47: 3. https://doi.org/10.1145/2693843

Petersen K, Feldt R, Mujtaba S, and Mattsson M (2008). Systematic mapping studies in software engineering. In the 12th International Conference on Evaluation and Assessment in Software Engineering, Bari, Italy, 12: 1-10. https://doi.org/10.14236/ewic/EASE2008.8

Pope S (1996). Application migration for mobile computers. In the Third International Workshop on Services in Distributed and Networked Environments, IEEE, Macau, Macau: 20-26. https://doi.org/10.1109/SDNE.1996.502443

Sajjad M, Abbasi AA, Malik A, Altamimi AB, and Alseadoon IM (2018). Classification and mapping of adaptive security for mobile computing. IEEE Transactions on Emerging Topics in Computing, 8(3): 814-832. https://doi.org/10.1109/TETC.2018.2791459

Sørensen CF, Wang AI, and Hoftun Ø (2003). Experience paper: migration of a web-based system to a mobile work environment. In the 21st IASTED International Multi-

Conference on Applied Informatics, Innsbruck, Austria: 1033-1038.

Sultan K (2019). Migration of existing software systems to mobile computing platforms: A systematic mapping study. Available online at: https://dspace.auk.edu.kw/handle/11675/5753

Williams BJ and Carver JC (2010). Characterizing software architecture changes: A systematic review. Information and Software Technology, 52(1): 31-51. https://doi.org/10.1016/j.infsof.2009.07.002

Winter A and Ziemann J (2007). Model-based migration to service-oriented architectures. Available online at: https://www.cs.vu.nl/csmr2007/workshops/2-%20winterziemann.pdf